Geo Brick LV and Micro-Stepping

Introduction

Traditional stepper-motor control systems are comprised of two separate types of components connected by wires. The first component is the "indexer", which performs the commanded trajectory generation. The second component is the "stepper drive", which receives the commanded position values from the indexer and applies the necessary currents to the phases of the stepper motor to try to achieve the commanded position.

The standard format of communication over wires between the indexer and the stepper drive(s) is "pulse and direction" – just two signals (although each is often provided on a differential signal pair). Each pulse from the indexer represents one position increment in the stepper drive, which could be a "full step", "half step", "mini-step" or "micro-step" depending on the drive.

The pulse-and-direction format has been popular because it keeps the number of wires between the devices to a minimum, but it does not make much sense for an integrated system like Geo Brick LV. The indexer must effectively differentiate the command position to produce an output pulse frequency, and the drive must effectively integrate the received signal using a counter. There is no need to do these mutually canceling steps in the hardware for an integrated system where the command value can be passed over a wide data bus.

Delta Tau calls this method of control of stepper motors "direct microstepping" to indicate that the commanded position value is passed directly to the phase control algorithm without intermediate pulse/differentiation and count/integration steps. Some refer to this technique as "vector control" of stepper motors because it has much in common with similar algorithms for control of induction motors.

Delta Tau's direct microstepping algorithm is simply one variant of its generalized motor phase commutation algorithm. Other variants use a measured sensor position to establish the rotor magnetic-field angle in order to decide how to apply current in the motor phases. Direct microstepping, by definition, does not use any rotor-angle sensors, and so uses commanded position instead (as "simulated feedback") to establish the rotor magnetic-field angle.

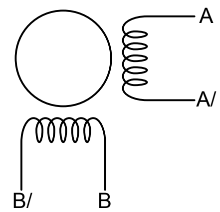
In PMAC products such as the Geo Brick LV, both closed-loop commutation and direct-microstepping open-loop commutation algorithms can operate on 2-phase or 3-phase motors. However, for historical reasons, the vast majority of motors used in closed-loop operation (e.g. brushless servomotors) are 3-phase, and the vast majority of used in open-loop operation (e.g. hybrid stepper motors) are 2-phase. This section will use 2-phase motors in its examples, but one a single variable (Ixx72) needs to be change for a 3-phase stepper motor.

Micro-Stepping Discription Step-by-Step

A stepper motor is very simple to control if one decides to just use the full step or half step approach. Most of the stepper motors available in the market either have 8 wires which user can select the coils to be either in series or parallel to achieve different characteristics or the manufacturer already has the coils pre-wired for use and there are a total of 4 wires available for the user. Both types are supported by the Geo Brick LV.

Take a look at the following table which explains the voltage change sequence for a bi-polar driver on a 4-wire stepper motor:

Terminal	Voltage Polarity CW→ ←CCW			
A	-	-	+	+
A/	+	+	-	-
B	-	+	+	-
B/	+	-	-	+

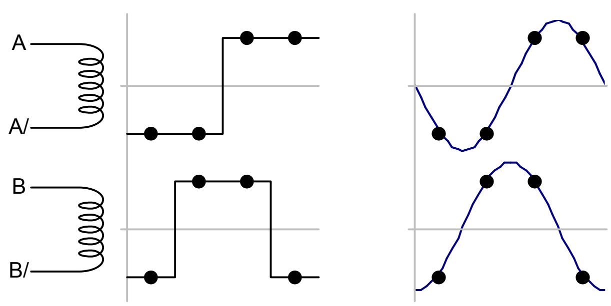


If the above sequence is repeated then the stepper will move 1 step in the direction of sequence. Based upon the previous table, you can see the current direction in each coil will be:

Coil	Voltage Polarity CW→ ←CCW			
A – A/	-	-	+	+
B – B/	-	+	+	-

Although this stepping can be achieved very easily, the resulting movement will be very rough. The roughness has two individual causes. The first one is the mechanical design of the stepper motors and the other one is the roughness caused by the sudden change in the current direction in each of the sequences mentioned above. If this change of current is turned into a more gradual change the motion of the stepper motor becomes smoother and higher resolutions can be achieved.

For a better understanding, the following graphs shows the current change in both full step method and Micro-Stepping method.



As shown in the graph on the right, the change of current can be gradual on the coils, which is two sine wave current shapes with a 90 degree phase offset (Ixx72 = 512) . This is exactly the approach that has been taken in Micro-Stepping. The current level control in done through the current loop calculations which transforms the current sensor readings on phases U and V into Q and D space (quadrature and direct current space) and closes the current loop on them, then transforms the results into PWM commands on Phases U and V. Depending on the initial setup of the Geo Brick LV amplifier, through the ADC strobe settings in initialization PLC, the phases W and X will output the corresponding voltages to achieve the desired voltage on the stepper coils.

The current level on each phase is dependent on the commutation angle. In a normal brushless motor this angle information is provided by an encoder. But for the stepper motors, which usually won't have a feedback device, another approach needs to be taken.

The main idea is to look at the IqCmd (Quadrature command of the motor) as a velocity command to the motor. Once integrated, the position can be obtained. This integration is done through the encoder conversion table. There are two entries for each stepper motor, first entry to read the IqCmd from the memory and the second entry to integrating the result.

	Motor 1	Motor 2	Motor 3	Motor 4	Motor 5	Motor 6	Motor 7	Motor 8
1st Entry (2 lines)	\$6800BF \$018018	\$68013F \$018018	\$6801BF \$018018	\$68023F \$018018	\$6802BF \$018018	\$68033F \$018018	\$6803BF \$018018	\$68043F \$018018
2nd Entry	\$EC00XX	\$EC00XX	\$EC00XX	\$EC00XX	\$EC00XX	\$EC00XX	\$EC00XX	\$EC00XX

The XX will represent the Bits 0-7, which form the fifth and sixth hex digits of the entry, specify the address offset from the beginning of the table to the first entry to be used, as an unsigned 8-bit quantity. The value in these digits should equal the number of the I-variable matching the first entry minus 8000.

There are 3 motor I-variables which point to the output of the ECT entry. The commutation position address (Ixx83), position loop feedback address (Ixx03) and velocity loop feedback address (Ixx04).

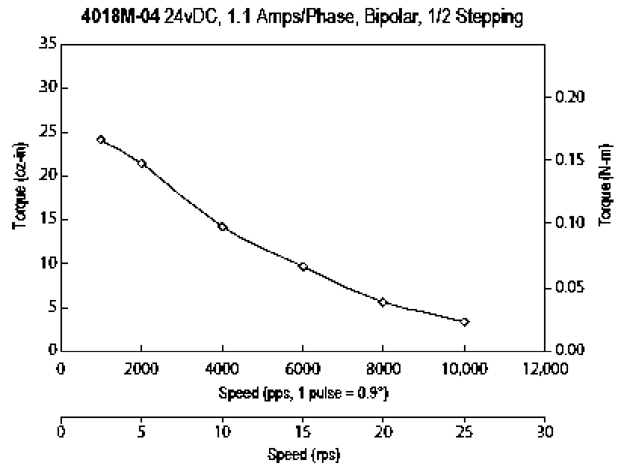
The Geo Brick LV's direct-microstepping algorithm provides 2048 microsteps per electrical cycle, which is 512 microsteps per full step on a 2-phase motor. The phase-commutation algorithm that implements direct microstepping is quite independent of the trajectory generation algorithm, so the two algorithms do not have to use the same units. However, most users will prefer to have the same unit resolution for both algorithms to eliminate confusion, so this section will specify setup variables such that one "microstep" in the phase-commutation algorithm will equal one "count" in trajectory-generation algorithms.

The next step is to define how many micro-steps we want per full step of the motor or in other words, how fine we want to generate the sine-wave shown earlier. In most drives available in market there are choices between a 256 micro-steps per full step and 512 micro-steps. Assuming 512 micro-step approach, we will have 2048 micro-steps per full sine-wave period since each sine-wave commutation covers 4 full steps. Since the commutation data is read through the ECT output, it would be in units of 1/32 of a count, which means the value which should be set as counts per N commutation cycle (Ixx71) equal to $2048 \times 32 = 65536$ while setting the number of commutation cycles to 1.

The next step is to establish a phase reference. Since the stepper motor is basically an open loop system, all we need to do is to set the commutation angle to a set number and once the amplifier is enabled, it will pull the motor to the same angle. This is achieved through setting the power-on phase position address (Ixx81) and power-on phase position format (Ixx91). By pointing the Ixx81 to the output of the ECT and settings the Ixx91=\$500000 (Read 16-bit parallel data from X memory) the last known phase position from the ECT is copied as the phase position on the motor.

There are two more parameters needs to be set before we are able to move the stepper motor in open loop mode. The first one is the magnetization current, which depends on the electrical characteristics of the motors and also the Ixx69 setting.

Ixx77, or what is called a magnetization current in PMAC, is actually the amplitude of the sine-wave current commutating the stepper. Higher current translates into higher torque output. In a normal stepper motor, the torque per unit current drops as the speed increases.



This can be compensated by increasing the Ixx77 as a function of motor speed through either a background PLC or an open servo routine. Ixx77 units would be the same as the current feedbacks. For Geo Brick LV the maximum value of 32767 in ADC feedbacks represents 33.85A, which means for the motor shown in the graph above the setting of Ixx77 would be

$$\text{Ixx77} = \frac{1.1^{Amps} \times 32767}{33.85^{Amps}} = 1064.8$$

Since the output of the PID loop is a velocity command for the motor (the position of the motor is based upon the integration of this command, IqCmd) the Ixx69 limit will control the maximum commanded speed (magnitude) in every servo cycle. The commutation position is being calculated in the ECT every servo cycle. For a successful commutation, six steps per commutation cycle should be calculated and commanded.

Electrical length based upon micro-stepping Ixx70=1 and Ixx71=65536

Maximum length allowed in commutation per servo cycle: $\frac{\text{Ixx71}}{6} = 10922.67$

This is basically the maximum value for IqCmd which is being integrated by the ECT. Since Ixx69 is limiting the upper 16 bits of IqCmd, every bit in Ixx69 represents 256 LSB in IqCmd.

$$\text{Ixx69} = \frac{10922.67}{256} = 42.66667$$

Once all these values are set and current loop is tuned, the open loop movement of the stepper motor is possible.

For quasi-closed loop control, since the feedback is generated internal to the PMAC, a set of preset values for the servo gains will always work for all the steppers.

Ixx30=1024

Ixx31=0

Ixx32=85

Ixx33=1024

Ixx34=1

Ixx35..xx39=0

Micro-Stepping Example Setup

In this section complete configuration for a single stepper motor on the first channel of a Geo Brick LV is included as reference and example for direct-microstepping algorithm.

Motor Information:

Model Number	Amp/Phase	Holding Torque oz-in	Holding Torque N-m	Resistance Ohm/Phase	Inductance mH/Phase	Inertia oz-in2	Weight Lbs.	Number of Leads
4018M-05	2.5	31.0	.22	0.5	0.8	0.13	0.48	4

```
I7100=1473          // setting PWM frequency to 20 kHz
I6800=1473          // =INT(((117964.8/PWM_Freq_kHz)-6)/4)
I7000=1473

I7101=3             // setting phase clock to 10 kHz
I6801=3             // Phase_Freq_kHz = (PWM_Freq_kHz*2)/(Phase_Divider+1)
I7000=3

I7102=1             // setting servo clock to 5 kHz
I6802=1             // Servo_Freq_kHz = Phase_Freq_kHz / (Servo_Divider+1)
I7002=1

I7104=1             // setting the PWM dead time to .135 usec for better
I7004=1             // PWM resolution close to zero output

I10=1677653         // = 117964.8 /((4*PWM_Freq_kHz+6) * (Phase_Divider+1)
                    //            *(Servo_Divider+1))

I5=2                // Background PLCs are permitted to execute

// PLC 1 sets up channel 1 for stepper motor. Refer to Geo Brick LV’s manual
// for complete settings

OPEN PLC 1 CLEAR
DISABLE PLC 2..31
I6111=500*8388608/I10 WHILE(I6111>0) ENDWHILE
I7006=$F84DFE                              // Setting Motor #1 Type
I6111=500*8388608/I10 WHILE(I6111>0) ENDWHILE
I7006=$F8CDFE                              // Setting Motor #1 CLRF
I6111=500*8388608/I10 WHILE(I6111>0) ENDWHILE
```

```
I7006=$F84DFE                                  // Setting Motor #1 Protection
I6111=500*8388608/I10 WHILE(I6111>0) ENDWHILE
DISABLE PLC 1
CLOSE

I8000=$6800BF       // Parallel X/Y read from $BF, IqCmd for Motor #1
I8001=$018018       // Reading upper 24 bits of the 48 bit word
I8002=$EC0001       // Reading the output of the previous entry and integrating
I8003=$0            // End of Table

I100=1              // Servo Activated
I101=1              // Commutation Enabled

I102=$078002        // first PWM output, IC0, channel 1, PWMA

I103=$3503          // Result of the integration in ECT
I104=$3503

I182=$078006        // Current loop feedback, IC0, Channel 1, ADCB
I184=$FFFC00        // 14-bit ADC feedback

I172=512            // 90 degree phase angle,

I166=1694           // PWM Scale Factor, = I7000 * 1.15

I161=0.02518        // Estimated current loop gains
I162=0              // Based upon resistance and inductance
I176=0.5785         // refer to Turbo Users Manual for calculations

I157=2963           // I2T Protection Based upon current limitations
I158=408            // refer to Turbo Users Manual for calculations

I169=42.6667        // In Direct Microstepping indicates the max speed

I125=$078000        // Flags Address, IC 0, Chan 1
I124=$820401        // High True Amp Fault, Over-travel Limits Disabled
                    // Disable 3rd Harmonic-Injection Algorithm

I183=$3503          // On-going Commutation pointing to ECT output

I170=1
I171=65536          // 512 counts per full step, 4 steps per commutation,
                    // 32 sub-counts on the output of ECT, = 512*4*32
I180=0
I173=0
I174=0

I181=$3503          // Absolute Phase Reference from ECT
I191=$500000        // Read lower 16 bits of ECT output (X word)

I177=987            // Current amplitude of the Sine-wave

I130=1024           // PID Gains
I131=0
I132=85
I133=1024
I134=1
I135=0
I136=0
I137=0
I138=0
I139=0
```